

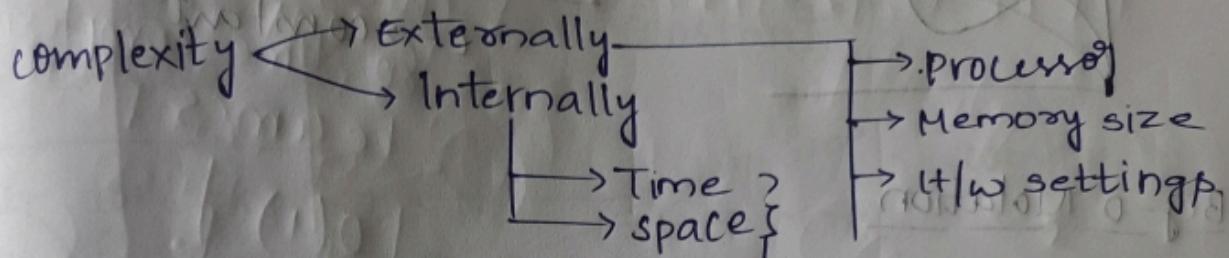
Data structures

Unit 8 Data Structures

10/08/22

(m) and i. Algorithms

complexity Analysis (m) i. (m)



Analysis

Q) (Asymptotic Notation Explain?) - (10m)

Time space trade off
only one is good satisfied

Asymptotic Notation

Omega Notation (Ω)

lower bounded values

Best case
(minimum no. of time)

Theta Notation (Θ)

tight bounded values

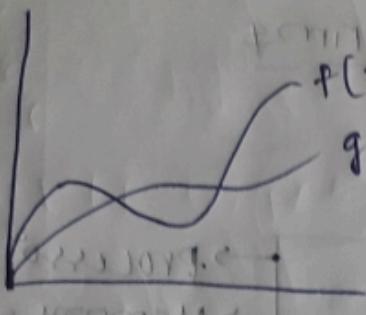
Average case

Big-O Notation (O)

upper bounded values

Worst case

Omega Notation



$$f(n) = g(n)$$

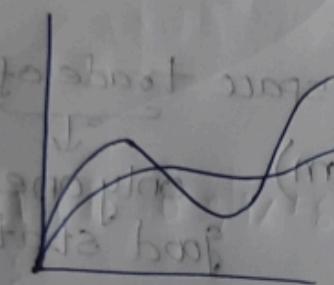
$$f(n) \geq c g(n)$$

$c \rightarrow \text{constant}$

\uparrow linear

\downarrow exponential

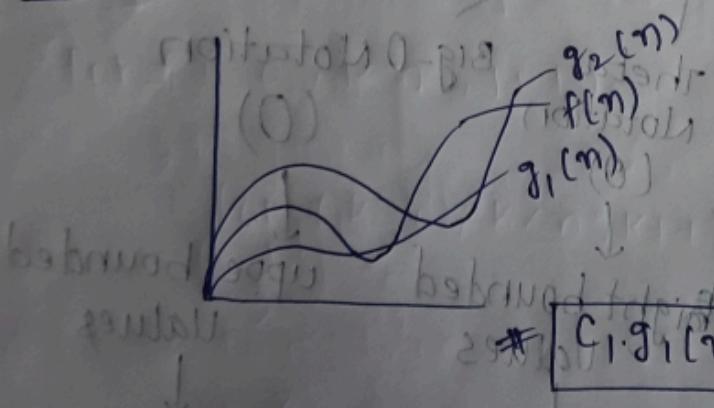
Big-O Notation



$$f(n) = g(n)$$

$$f(n) \leq c g(n)$$

Theta Notation



$$f(n), g_1(n), g_2(n), \dots$$

$$f(n) = g(n)$$

$$c_1 g_1(n) \leq f(n) \leq c_2 g_2(n)$$

(min & max)

- Rules of Big-O Notation to find time complexity
- 1) Nested loops are multiplied together
 - 2) Sequential loops are added
 - 3) Larger term will be kept and smaller term will drop.
 - 4) Conditional checks are constant.
 - 5) Constants are dropped.

Nested loop

for ($i=0; i < n; i++$) → $O(n)$ } very imp
 for ($j=0; j < n; j++$) → $O(n)$ } complexity is $O(n^2)$ bcz → Rule ①

sequential loops

for ($i=0; i < n; i++$) → $O(n)$ } complexity is $O(n) + O(n)$
 for ($j=0; j < n; j++$) → $O(n)$ } $n + n = O(2n)$ bcz → Rule ②
 finally complexity is $O(n)$
 (2 is dropped bcz it is constant → Rule 5)

(2)

3) Use pd (multiplication)

3) Have its proof

37) $\text{for } (i=0; i < 58; i++)$ 58 5

$\{$ This loop will run 58 times

$\}$ These conditions are constant so time complexity becomes

$\text{for } (i=0; i < 5; i++)$ This loop will run 5 times

$\{$ These conditions are constant so time complexity becomes

$\boxed{O(1)}$

Rule ④

47) $\text{for } (i=0; i < 58; i++)$ 58 1

$\{$ This loop will run 58 times

$\}$ Time complexity is O(1)

$\text{for } (j=0; j < n; j++)$ n

$\{$ This loop will run n times

$\}$ Time complexity is O(n)

$\text{for } (k=0; k < n; k++)$ n

$\{$ This loop will run n times

$\}$ Time complexity is O(n^2)

$\text{for } (v=0; v < n; v++)$ n

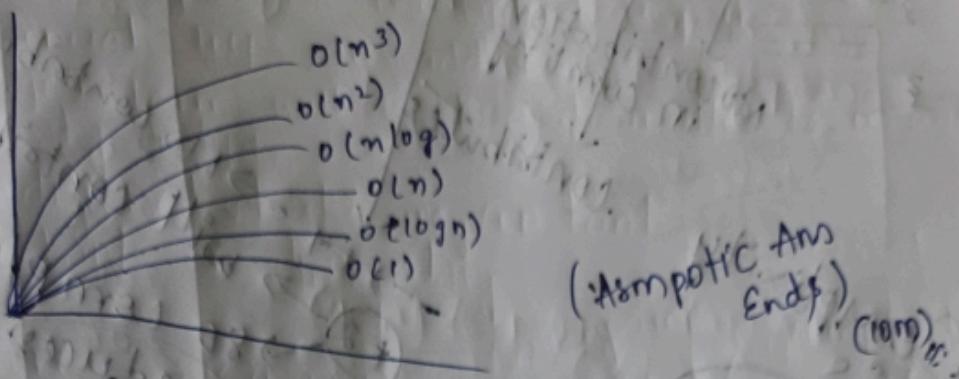
$\{$ This loop will run n times

$\}$ Time complexity is O(n)

Complexity is $O(n^2)$ by Rule ③
i.e. Large kept smaller leave

IMP

Graph



① for (int $i=0$; $i < n$; $i++$) $\rightarrow O(n)$

for ($i=0$; $j < n$; $j++$) $\rightarrow O(n) \Rightarrow O(n^2)$

② for ($i=0$; $i < 1000000$; $i++$) $\rightarrow O(1)$

③ for ($i=0$; $i < m$; $i++$) $\rightarrow O(n)$

for ($i=0$; $i < m$; $i++$) $\rightarrow O(n) \rightarrow O(nm)$

for ($j=0$; $j < m$; $j++$) $\rightarrow O(m)$

④ for ($i=0$; $i < n$; $i++$) $\rightarrow O(n)$

for ($i=0$; $j < n$; $j++$) $\rightarrow O(\frac{n(n+1)}{2})$

$O(\frac{n(n+1)}{2})$ depends
on outerloop

$\frac{n^2}{2}$ depends
on count

$O(nxht)$

6) for (int $i=0$; $i < n$; $i^* = 2$)
 {
 }
 No contribution
 in complexity
 contributes
 \rightarrow no contribution
 $\star 2 \rightarrow$ will contribute
 so complexity
 reduces
 Complexity
 is
 $O(\log n)$
 $\boxed{O(\log n)}$
 $b60z$ see
 graph
 $O(n) > O(\log n)$

7) for (int $i=0$; $i < n$; $i++$) $\rightarrow O(n)$
 {
 }
 \rightarrow adding or removing
 elements

for (int $i=0$; $i < n$; $i^* = 2$)
 {
 }
 \rightarrow complexity
 $O(n \log n)$

Complexity

$\boxed{O(n \log n)}$

$P_1 \rightarrow 3n^4 + 2n^3 + 15n^2 + 2n + 1$
 $P_2 \rightarrow 2n + 1$
 $P_3 \rightarrow 15n^2 + 4n - 1$
 $\rightarrow O(n^4)$

For polynomials
 complexity is highest degree

$O(n^4) \downarrow$
 $O(n)$
 $O(n^2)$

complexity is
 $\rightarrow O(n^4)$

11/08/2022
Array

* Length (No. of Elements in array) = $[UB - LB + 1]$

Ex :- $A[-5, 10] \Rightarrow 10 - (-5) + 1 = 10 + 6 = 16$ elements present.

* Representation of Linear Array in Memory:-

$$LOC[LA[K]] = \text{Base}(LA) + W(K - LB)$$

$W \Rightarrow$ size of datatype.

Ex :- Address at loc $\rightarrow 6$

$$\text{Ans} \rightarrow LOC(LA(6)) = 200 + 1(6-0) = 200 + 6 = 206$$

Q:- Find the address for $LA[6]$ Each element of the array occupy 1 byte

200	LA[0]
201	
202	
203	
204	
205	
206	
207	
208	
209	
210	
211	
212	
213	
214	
215	
216	
217	
218	
219	
220	
221	
222	
223	
224	
225	
226	
227	
228	
229	
230	
231	
232	
233	
234	
235	
236	
237	
238	
239	
240	
241	
242	
243	
244	
245	
246	
247	
248	
249	
250	
251	
252	
253	
254	
255	
256	
257	
258	
259	
260	
261	
262	
263	
264	
265	
266	
267	
268	
269	
270	
271	
272	
273	
274	
275	
276	
277	
278	
279	
280	
281	
282	
283	
284	
285	
286	
287	
288	
289	
290	
291	
292	
293	
294	
295	
296	
297	
298	
299	
300	
301	
302	
303	
304	
305	
306	
307	
308	
309	
310	
311	
312	
313	
314	
315	
316	
317	
318	
319	
320	
321	
322	
323	
324	
325	
326	
327	
328	
329	
330	
331	
332	
333	
334	
335	
336	
337	
338	
339	
340	
341	
342	
343	
344	
345	
346	
347	
348	
349	
350	
351	
352	
353	
354	
355	
356	
357	
358	
359	
360	
361	
362	
363	
364	
365	
366	
367	
368	
369	
370	
371	
372	
373	
374	
375	
376	
377	
378	
379	
380	
381	
382	
383	
384	
385	
386	
387	
388	
389	
390	
391	
392	
393	
394	
395	
396	
397	
398	
399	
400	
401	
402	
403	
404	
405	
406	
407	
408	
409	
410	
411	
412	
413	
414	
415	
416	
417	
418	
419	
420	
421	
422	
423	
424	
425	
426	
427	
428	
429	
430	
431	
432	
433	
434	
435	
436	
437	
438	
439	
440	
441	
442	
443	
444	
445	
446	
447	
448	
449	
450	
451	
452	
453	
454	
455	
456	
457	
458	
459	
460	
461	
462	
463	
464	
465	
466	
467	
468	
469	
470	
471	
472	
473	
474	
475	
476	
477	
478	
479	
480	
481	
482	
483	
484	
485	
486	
487	
488	
489	
490	
491	
492	
493	
494	
495	
496	
497	
498	
499	
500	

Q. Find the address for LA[15].
Each element of the array
occupy 2 bytes?

Sol:- $LA(15) = 200 + (2)(15 - 0)$
 $= 200 + 30$
 $LA(15) = 230$

200	L(0)
201	
202	L(1)
203	
204	L(2)
205	
206	L(3)
207	

* Algorithm for traversing an array.

17/August,

* In sertion and Deleting to database

* Insertion : Adding an element

- Beginning

- Middle

- End

* Deletion : Removing an element

- Beginning

- Middle

- End

→ Time complexity

Insertion
Deletion at Beg $\Rightarrow O(1)$

Insertion
Deletion at End $\Rightarrow O(n)$

If array
is
empty

L(0) Insertion at beginning = $O(n)$ \Rightarrow If array is filled fully
Deletion

L(1) Insertion at middle = $O(n)$
Deletion

* Linear search

Att \rightarrow 5M

CA \rightarrow 15 M \rightarrow Harker the platform

Term (MCQ + subjective) \rightarrow 40M

CA Mode

1) Practice problems \rightarrow 30M

2) coding Test \rightarrow 15M (After mid)

3) Online Quiz \rightarrow 10M

15th October 2022 \rightarrow in Hackerrank

Best ① out 2

Q v/s ① \rightarrow 10th September

Quiz ② \rightarrow 5th November